University of Saskatchewan
Department of Computer Science

Cmpt 330
C Programming Quiz

February 4, 2002

Time: 50 minutes
Total Marks: 50

Professor: A. J. Kusalik
Closed Book

Name: ______________________________________

Student Number: ____________________

Directions:

Answer each of the following questions in the space provided in this exam booklet. If you must continue an answer (e.g. in the extra space on the last page, or on the back side of a page), make sure you clearly indicate that you have done so and where to find the continuation.

Make all written answers legible; no marks can be given for answers which cannot be decrypted. Where a discourse or discussion is called for, be concise and precise.

Use of calculators is not allowed during the exam. Fortunately, you should not need a calculator for completing any of the questions. The last page of the exam contains supplemental information which may be of use in answering some of the questions.

The context for questions is the C programming language as presented in the "C Short Course" given as part of Cmpt330. If you find it necessary to make any assumptions to answer a question, state the assumption with your answer. Note that answers involving constructs in C++, but not C, will not be considered correct.

Marks for each major question are given at the beginning of that question. There are a total of 50 marks (one mark per minute).

Good luck.

For marking use only:

A.	_____/9	D.	_____/6
B.	_____/8	E.	_____/13
C.	_____/8	F.	_____/6

Total: ______/50

A. *(1 mark each, for 9 marks)*

For each of the statements below, indicate whether it is **true** ("T") or **false** ("F").

___ The following two statements are equivalent (the storage class and type of variable a is the same in each case):

```
char a[] = "C World!";
```

and

```
char *a = "C World!";
```

___ A `static` variable not explicitly initialized is automatically initialized by default as if it (or its members, if it is an array) were assigned the constant 0.

___ On a machine with a word size of 32 bits, variables of type `long int` and `(int *)` are of the same size.

___ In C, arguments which are non-pointer types are passed to a function using call-by-value. Arguments which are pointer types are passed by reference.

___ A function prototype defines the number and types of arguments which are passed to a function, and the return type of the function (type of the return value, if any).

___ Comments can be nested; that is, one piece of comment (delimited by "`/*`" and "`*/`") can appear within another (also delimited by "`/*`" and "`*/`").

___ The escape sequence for a carriage-return character is `\r`.

___ The *`lint(1)`* command is used to remove extra or extraneous white space from a C program.

___ To create a `.h` file (header file) from a normal `.c` file, one invokes the C compiler (*cc*) with the `-h` option. For instance,

```
        cc -h mystdio.c
```

will create `mystdio.h` from `mystdio.c`

B. *(3+2+3 = 8 marks)*

Answer each of the following questions with a concise, precise answer.

1. What is the meaning of each of the following operators? I.e. say what operation, or combination of operations, each of the following operators performs.

(a) `&`

(b) `<<=`

2. Name two unconditional branch statements (constructs).

3. Write a program segment equivalent to the following which uses a `while`-statement instead of the `for`-statement. Your program segment must perform the same operations in the same order as the given code.

```
int i, j, num[20], sqr[20];
for (i=0, j=0; i < 20; i++, j++){
    num[i] = i;
    sqr[i] = i*j;
}
```

C. ***(8 marks)***

Suppose we have variables declared and assigned as shown below:

```
int i, j, k;
float x, y;

i = j = k = 3;
x = 0.0;
y = 2.3;
```

Complete the following table. An example is given to illustrate what needs to be provided to complete the table. The "equivalent expression" must make explicit, by use of parenthesis, the precedence of all operators. Assume that, prior to evaluation of each expression, the values of the variables are as initialized above.

Expression	Equivalent Expression	Value
`i && j && k`	`(i && j) && k`	1
`y || i && j - 3`		
`i---i-j`		
`i * j && x < y`		
`x != y && j + 1 == !k + 4`		

D. (3+3 = 6 marks)

Knowledge about C programming concepts is necessary to prevent errors when programming. For each of the following C code segments, indicate and describe the error being made. I.e. in each of the following pieces of code, a programming error is being made. Identify (in some easily discernible way) the error in each case. Also describe (by way of a short description) the nature of the error. Then indicate how the error can be concisely corrected without changing the intended logic of the program sample.

1.

```
#define NAME "Jane Doe"

int main( void ) {
{
    char *s;

    *s = NAME;
    printf("%s\n", NAME);
    printf("%s\n", s);
    return 0;
}
```

Nature of the error:

2.

```
#include <stdio.h>
#include <string.h>

main()
{
    File *infile;
    int x;

    infile = fopen( "foobar", "r" );
    fscanf( infile, "%d", &x );
    printf( "the number is %x", x );
}
```

Nature of the error:

E. (3+1+2+3+3+1 = 13 marks)

Suppose that you are writing a C program to process a simple student information database, and produce output reports. The input file to be used by your program has up to 100 rows of data, one for each student. Each row contains six fields of data, namely the student's student number (an integer), last name (maximum of 10 letters), first name (maximum of 10 letters), college (2 letters), year of program (an integer), and cumulative grade point average (CGPA) (a rational number), in that order. For example, the first few lines of the input file might look like this:

```
123456 Smith Chris      AR  3 84.0
192765 Balloo Mogley    AR  4 74.9
612076 Chang Tzue       EN  4 81.2
742806 Chretien Jeanne  CO  3 87.1
812712 Baggins Bilbo    GS 15 64.6
904615 Wang Yanwei      EN  4 91.0
```

Assume that your program must read in and store this information for subsequent processing (e.g., sorting by last name or by CGPA).

(a) Below, show the declaration of a `struct StudentRecord` that is capable of storing one row of the above information.

(b) In the space below, define the manifest constant `MAX_STUDENTS` which is the maximum number of `StudentRecord` structures that may be necessary (in your program) to hold all the data in the input file.

(c) Show the declaration of an array, `studentlist`, of `StudentRecord` structures (assuming the `StudentRecord` type as declared in part (a)). The array `studentlist` is capable of storing up to `MAX_STUDENTS` student records.

(d) Show the declaration of a variable `studentrecp` which is a pointer to a record of type `StudentRecord` (defined in part (a)). Then show the allocation of a `StudentRecord struct` from dynamic-allocated memory, and the assignment to `studentrecp` of the pointer to this memory. Make sure there are no type errors and make all type castings explicit.

(e) Show the declaration of a variable studentreclist which is a pointer to a dynamically-allocated array of pointers, each of which points to a StudentRecord struct. Then show the statement necessary to allocate storage for an array of size MAX_STUDENTS, where each element of the array is a pointer to a StudentRecord struct, and assign to studentreclist the pointer to the dynamically-allocated storage. Make sure there are no type errors and make all type castings explicit. Finally, make the 0th row of studentreclist be the record pointed to by studentrecp (from part (d)).

(f) Suppose that records for students have been read in from the file, and information stored in dynamically-allocated storage pointed to by studentreclist. Suppose that c is declared to be of type char. Give a statement which will assign to c the value of the 3rd character of the last name of the fifth student. Remember that in C indexing begins at 0.

F. (2+4 = 6 marks)

Answer each of the following questions with a concise answer.

1. Give two reasons why the C language became extremely popular, and continues to be popular more than 25 years after its inception.

2. Give at least two advantages and two disadvantages of using macros instead of functions. You can use examples to help make your point(s) clear.

Supplementary information

You may find the following function prototypes useful in answering some of the questions in this exam:

```
int fclose( FILE *stream );
int fgetc( FILE *stream );
char *fgets( char *s, int size, FILE *stream );
FILE *fopen( const char *path, const char *mode );
int fprintf( FILE *stream, const char *format, ... );
int fputc( int c, FILE *stream );
int fputs( const char *str, FILE *stream );
void free( void *ptr );
int getc( FILE *stream );
int getchar();
char *gets( char *str );
void *malloc( size_t size );
int open( const char *path, int flags, mode_t mode);
void perror( const char *string );
int printf( const char *format, ... );
int putc( int c, FILE *stream );
int putchar( int c );
int puts( const char *str );
void *realloc( void *ptr, size_t size );
int scanf( const char *format, ... );
int sprintf( char *str, const char *format, ...);
```

Extra Space

(The space below is for answering previous questions or for rough work.)